

This Page Is Inserted by IFW Operations
and is not a part of the Official Record

BEST AVAILABLE IMAGES

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images may include (but are not limited to):

- BLACK BORDERS
- TEXT CUT OFF AT TOP, BOTTOM OR SIDES
- FADED TEXT
- ILLEGIBLE TEXT
- SKEWED/SLANTED IMAGES
- COLORED PHOTOS
- BLACK OR VERY BLACK AND WHITE DARK PHOTOS
- GRAY SCALE DOCUMENTS

IMAGES ARE BEST AVAILABLE COPY.

As rescanning documents *will not* correct images,
Please do not report the images to the
Image Problem Mailbox.

PATENT ABSTRACTS OF JAPAN

(11)Publication number : 11-053187

(43)Date of publication of application : 26.02.1999

(51)Int.Cl.

G06F 9/30
G06F 9/305
G06F 9/38

(21)Application number : 09-204125

(71)Applicant : MATSUSHITA ELECTRIC IND CO LTD

(22)Date of filing : 30.07.1997

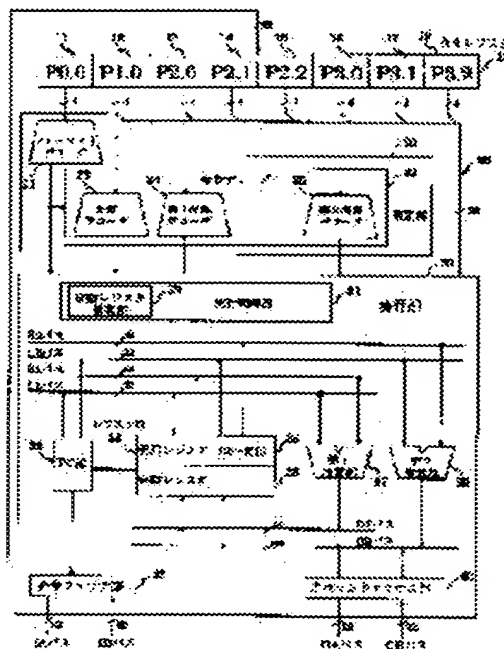
(72)Inventor : HEIJI TAKEHITO
HIGAKI NOBUO
TANAKA AKIRA
TANAKA TETSUYA
TAKAYAMA SHUICHI
KOTANI KENSUKE
MIYAJI SHINYA

(54) PROCESSOR

(57)Abstract:

PROBLEM TO BE SOLVED: To decrease the number of execution cycles by eliminating an overhead due to separation of operation for accumulating a constant and operation for using it by using even a 1st constant stored in a constant register in combination when a 2nd constant is used as an operand.

SOLUTION: A decoding part 20 decodes an instruction held in an instruction register 10 and outputs the control line corresponding to the decoding result to an execution part 30. When a state wherein a 1st constant is placed in the instruction is decoded, the 1st constant is stored in a constant register 36 and when a state wherein a 2nd constant is placed is decoded, the constant stored in the constant register 36 is read out and combined with the 2nd constant to obtain an operand. Therefore, while a constant as an operand is generated from the constant accumulated in the constant register 36 and the constant specified by operation, the constant is used to perform operation.



LEGAL STATUS

[Date of request for examination]

09.04.2001

[Date of sending the examiner's decision of rejection]

[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]

[Date of final disposal for application]

[Patent number]

[Date of registration]

[Number of appeal against examiner's decision
of rejection]

[Date of requesting appeal against examiner's
decision of rejection]

[Date of extinction of right]

Copyright (C); 1998,2003 Japan Patent Office

(19)日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11)特許出願公開番号

特開平11-53187

(43)公開日 平成11年(1999)2月26日

(51)Int.Cl.⁹
G 0 6 F 9/30
9/305
9/38
識別記号
3 5 0
3 7 0

F I
G 0 6 F 9/30
9/38
9/30
3 5 0 F
3 7 0 X
3 4 0 F

審査請求 未請求 請求項の数10 O L (全 22 頁)

(21)出願番号 特願平9-204125
(22)出願日 平成9年(1997)7月30日

(71)出願人 00005821
松下電器産業株式会社
大阪府門真市大字門真1006番地
(72)発明者 瓶子 岳人
大阪府門真市大字門真1006番地 松下電器
産業株式会社内
(72)発明者 楢垣 信夫
大阪府門真市大字門真1006番地 松下電器
産業株式会社内
(72)発明者 田中 旭
大阪府門真市大字門真1006番地 松下電器
産業株式会社内
(74)代理人 弁理士 滝本 智之 (外1名)

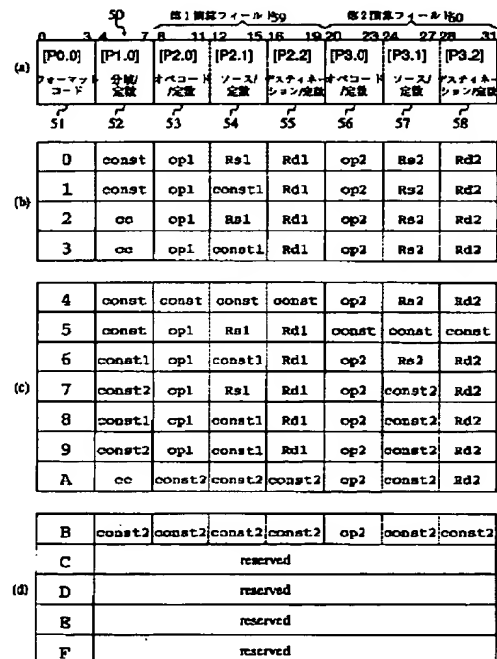
最終頁に続く

(54)【発明の名称】 プロセッサ

(57)【要約】

【課題】 命令を実行するプロセッサにおいて、命令中の無駄領域を後続命令で使用する定数を分割して埋めるために実装される定数レジスタの値の、効率的な蓄積・利用方法により、実行サイクル数の低減を図る。

【解決手段】 上記定数レジスタに蓄積された定数とオペレーションで指定された分割された定数とを連結してオペランドとする定数を生成すると同時に、その定数をオペランドとしてオペレーションを実行することを可能とする。さらに、上記定数レジスタの内容を参照するのみで、定数レジスタの内容を変化させないオペレーションを備える。また、上記定数レジスタの内容の一部のみを変更し、残りの部分の内容を変化させないオペレーションを備える。



【特許請求の範囲】

【請求項1】 複数の命令を格納する記憶装置から命令を読み出す命令読み出し手段と、

定数を格納するための定数レジスタと、

前記命令中に第1の定数が置かれていることと、前記命令中に第2の定数が置かれていることを解読する解読手段と、

前記解読手段により前記第1の定数が置かれていると解読された場合に、前記第1の定数を前記定数レジスタに格納する格納手段と、

前記解読手段により前記第2の定数が置かれていると解読された場合に、前記定数レジスタに格納されている定数を読み出し、その読み出された定数と前記第2の定数とを連結した定数をオペランドとするオペレーションを実行する実行手段とを備えることを特徴とするプロセッサ。

【請求項2】 複数の命令を格納する記憶装置から命令を読み出す命令読み出し手段と、

定数を格納するための定数レジスタと、

前記命令中に第1の定数が置かれていることと、前記命令中に第2の定数が置かれていることを解読する解読手段と、

前記解読手段により前記第1の定数が置かれていると解読された場合に、前記定数レジスタに定数が格納されていないと判断すると前記第1の定数を所定位置に格納し、前記定数レジスタに定数が既に格納されていると判断すると前記定数レジスタに既に格納されている定数を残したまま前記第1の定数を新たに前記定数レジスタに格納する格納手段と、

前記解読手段により前記第2の定数が置かれていると解読された場合に、前記定数レジスタに格納されている定数を読み出し、その読み出された定数と前記第2の定数とを連結した定数をオペランドとするオペレーションを実行する実行手段とを備えることを特徴とするプロセッサ。

【請求項3】 前記解読手段は、さらに、前記命令中に所定のオペレーションが置かれていることを解読し、

前記実行手段は、

前記定数レジスタに格納されている定数を読み出し、その読み出された定数と前記第2の定数とを連結した定数をオペランドとする第1のオペレーションと、読み出された定数のみをオペランドとする第2のオペレーションとを選択的に実行し、

前記解読手段により前記第2の定数が置かれていると解読された場合には前記第1のオペレーションを実行し、前記解読手段により前記所定のオペレーションが置かれていると解読された場合には、前記所定のオペレーションを前記第2のオペレーションとして実行することを特徴とする請求項1または2記載のプロセッサ。

【請求項4】 複数の命令を格納する記憶装置から命令

を読み出す命令読み出し手段と、

定数を格納するための定数レジスタと、

前記命令中に定数が置かれていることを解読する解読手段と、

前記解読手段により定数が置かれていると解読された場合に、前記定数レジスタに定数が格納されていないと判断すると前記定数を所定位置に格納し、前記定数レジスタに定数が既に格納されていると判断すると前記定数レジスタに既に格納されている定数を残したまま前記定数を新たに前記定数レジスタに格納する格納手段と、

前記定数レジスタに格納されている定数を読み出し、その定数をオペランドとするオペレーションを実行する実行手段と、

前記実行手段により前記定数レジスタから定数が読み出されたことを示す状態保持手段とを備え、

前記格納手段は、

前記状態保持手段により定数が読み出されたことを示されていると、前記定数を前記定数レジスタに格納する前に前記定数レジスタの内容を消去することを特徴とするプロセッサ。

【請求項5】 前記プロセッサは、さらに、

前記実行手段により前記定数レジスタから定数が読み出されたことを示す状態保持手段を備え、

前記格納手段は、

前記状態保持手段により定数が読み出されたことを示されていると、前記第1の定数を前記定数レジスタに格納する前に前記定数レジスタの内容を消去することを特徴とする請求項1から請求項3いずれか記載のプロセッサ。

【請求項6】 前記実行手段は、

前記状態保持手段により定数が読み出されたことを示されていると、前記定数レジスタに格納されている定数を繰り返し読み出し、その定数の少なくとも一部をオペランドとするオペレーションを実行することを特徴とする請求項4または請求項5記載のプロセッサ。

【請求項7】 前記解読手段は、さらに、前記命令中に第3の定数を伴う定数置換オペレーションが置かれていることを解読し、

前記格納手段は、前記解読手段により前記定数置換オペレーションが置かれていると解読された場合に、前記定数レジスタに格納されている定数の一部を残したまま他の部分と前記第3の定数とを置換して格納することを特徴とする請求項1から請求項3いずれか記載のプロセッサ。

【請求項8】 前記置換する部分が、前記定数レジスタの最下位ビットを含む部分である請求項7記載のプロセッサ。

【請求項9】 複数の命令を格納する記憶装置から命令を読み出す命令読み出し手段と、

定数を格納するための定数レジスタと、

10

20

30

40

50

前記命令中に第1の定数が置かれていることと、前記命令中に第2の定数が置かれていることを解読する解読手段と、

前記解読手段により前記第1の定数が置かれていると解読された場合に、前記定数レジスタに定数が格納されていないと判断すると前記第1の定数を所定位置に格納し、前記定数レジスタに定数が既に格納されていると判断すると前記定数レジスタに既に格納されている定数を残したまま前記第1の定数を新たに前記定数レジスタに格納する格納手段と、

前記実行手段により前記定数レジスタから定数が読み出されたことを示す状態保持手段と、

前記解読手段により前記第2の定数が置かれていると解読された場合であって、

前記状態保持手段により定数が読み出されていないことを示されている場合は、前記定数レジスタに格納されている定数を読み出し、その読み出された定数と前記第2の定数とを連結した定数をオペランドとするオペレーションを実行し、

前記状態保持手段により定数が読み出されたことを示されている場合は、前記第2の定数のみをオペランドとするオペレーションを実行する実行手段とを備えることを特徴とするプロセッサ。

【請求項10】 命令のフォーマットを指定するフォーマットコードが置かれるフォーマットフィールドと、並列実行させるオペレーションを指定する複数のオペレーションフィールドとを含む命令を解読し実行するVLIW方式のプロセッサであって、
解読手段は、前記フォーマットコードを参照することにより、少なくとも1つの前記オペレーションフィールドに定数が置かれていることを解読することを特徴とする請求項1から請求項9いずれか記載のプロセッサ。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】本発明は、マイクロプロセッサに関し、特に実行サイクル数の減少を図る技術に関する。

【0002】

【従来の技術】近年のマイクロプロセッサ応用製品の高機能化及び高速化に伴い、コード効率の高いプログラムを実行することができるマイクロプロセッサ（以下、単に「プロセッサ」という。）が望まれている。つまり、プログラムを構成する各命令中には無駄なコードや未使用の領域が含まれないことが望ましい。

【0003】ところが、特に、VLIW（Very Long Instruction Word）の如く固定長命令の場合には、命令中にノーオペレーションコード（nopコード）等の無駄なコードを置く必要が生じる。VLIWは複数のオペレーションフィールドからなり、各オペレーションフィールドにおいてプロセッサが備える複数の演算ユニットに

対応したオペレーションが指定されるが、オペレーションの依存関係等により、常に並列実行可能な複数のオペレーションが存在するとは限らないからである。

【0004】このようなnopコードの挿入に伴うコード効率の低下を回避する従来の技術として、例えば、特開平8-161169に開示されたVLIW方式の計算機システムがある。これは、nopコードを配置する必要があるオペレーションフィールドに、他の命令で用いられる定数を配置しておき、その配置情報を参照して配置してある定数を他の命令のオペランドとして用いる、というシステムである。

【0005】しかしながら、上記のシステムでは、無駄領域を埋めることができる定数の大きさが制限されるという問題がある。つまり、オペレーションフィールド長よりも長い定数を配置することができない。そこで、もう一つの従来技術として、特願平9-159049において出願人は、nopコードを配置する必要があるオペレーションフィールドに分割した定数を配置し、その定数を内部の定数レジスタに蓄積していき、他の命令でその定数レジスタを指定することにより蓄積した定数をその命令のオペランドとして用いる、というシステムを提案している。このようにして、後続命令で使用される定数を分割して配置していくことにより、命令中にnopコードを挿入することによる無駄領域の発生を回避している。

【0006】

【発明が解決しようとする課題】しかしながら、上記の定数を内部の定数レジスタに蓄積していくシステムでは、定数レジスタに定数を蓄積するオペレーションと、定数レジスタに蓄積された定数を使用するオペレーションが分離しており、定数を使用するオペレーションで定数レジスタを指定する必要があるため、必要以上に実行サイクル数が増大するという問題点を有している。

【0007】また、上記のシステムでは、定数レジスタに蓄積された定数を一度使用すると、次にその定数と全く同じ値や一部しか異ならない値を蓄積する場合でも、一から定数レジスタのすべてのビットを蓄積する必要があり、実行サイクル数が増大するという問題点を有している。そこで、本発明はかかる問題点に鑑みてなされたものであり、定数レジスタに定数を蓄積するオペレーションと、定数レジスタの定数を使用するオペレーションが分離していることによるオーバーヘッドを解決し、実行サイクル数を低減することを目的とする。

【0008】さらに、定数レジスタに蓄積された定数を一度しか使用できない点を改善し、定数レジスタの内容の一部又は全部を繰返して読み出せるようにすることにより、実行サイクル数を低減することを目的とする。

【0009】

【課題を解決するための手段】上記目的を達成するために請求項1記載のプロセッサは、複数の命令を格納する

10

20

30

40

50

記憶装置から命令を読み出す命令読み出し手段と、定数を格納するための定数レジスタと、前記命令中に第1の定数が置かれていることと、前記命令中に第2の定数が置かれていることを解読する解読手段と、前記解読手段により前記第1の定数が置かれていると解読された場合に、前記第1の定数を前記定数レジスタに格納する格納手段と、前記解読手段により前記第2の定数が置かれていると解読された場合に、前記定数レジスタに格納されている定数を読み出し、その読み出された定数と前記第2の定数とを連結した定数をオペランドとするオペレーションを実行する実行手段とを備えることを特徴とする。

【0010】これによって、第2の定数がオペランドとして使用される際に定数レジスタに格納された第1の定数も併せて使用される。また上記目的を達成するために請求項2記載のプロセッサは、複数の命令を格納する記憶装置から命令を読み出す命令読み出し手段と、定数を格納するための定数レジスタと、前記命令中に第1の定数が置かれていることと、前記命令中に第2の定数が置かれていることを解読する解読手段と、前記解読手段により前記第1の定数が置かれていると解読された場合に、前記定数レジスタに定数が格納されていないと判断すると前記第1の定数を所定位置に格納し、前記定数レジスタに定数が既に格納されていると判断すると前記定数レジスタに既に格納されている定数を残したまま前記第1の定数を新たに前記定数レジスタに格納する格納手段と、前記解読手段により前記第2の定数が置かれていると解読された場合に、前記定数レジスタに格納されている定数を読み出し、その読み出された定数と前記第2の定数とを連結した定数をオペランドとするオペレーションを実行する実行手段とを備えることを特徴とする。

【0011】これによって、複数の命令に跨って分割配置されていた第1の定数の断片が定数レジスタに格納されて蓄積され、元の定数に復元される。さらに第2の定数がオペランドとして使用される際に定数レジスタに格納復元された第1の定数も併せて使用される。ここで請求項3記載のプロセッサは、請求項1または2記載のプロセッサにおいて、前記解読手段は、さらに、前記命令中に所定のオペレーションが置かれていることを解読し、前記実行手段は、前記定数レジスタに格納されている定数を読み出し、その読み出された定数と前記第2の定数とを連結した定数をオペランドとする第1のオペレーションと、読み出された定数のみをオペランドとする第2のオペレーションとを選択的に実行し、前記解読手段により前記第2の定数が置かれていると解読された場合には前記第1のオペレーションを実行し、前記解読手段により前記所定のオペレーションが置かれていると解読された場合には、前記所定のオペレーションを前記第2のオペレーションとして実行するとしたものである。

【0012】また上記目的を達成するために請求項4記

載のプロセッサは、複数の命令を格納する記憶装置から命令を読み出す命令読み出し手段と、定数を格納するための定数レジスタと、前記命令中に定数が置かれていることを解読する解読手段と、前記解読手段により定数が置かれていると解読された場合に、前記定数レジスタに定数が格納されていないと判断すると前記定数を所定位置に格納し、前記定数レジスタに定数が既に格納されていると判断すると前記定数レジスタに既に格納されている定数を残したまま前記定数を新たに前記定数レジスタに格納する格納手段と、前記定数レジスタに格納されている定数を読み出し、その定数をオペランドとするオペレーションを実行する実行手段と、前記実行手段により前記定数レジスタから定数が読み出されたことを示す状態保持手段とを備え、前記格納手段は、前記状態保持手段により定数が読み出されたことを示されていると、前記定数を前記定数レジスタに格納する前に前記定数レジスタの内容を消去することを特徴とする。

【0013】これによって、複数の命令に跨って分割配置されていた定数の断片が定数レジスタに格納されて蓄積され、元の定数に復元される。復元された定数は、読み出された後に次に新たな定数の断片が定数レジスタに格納される前に消去される。ここで請求項5記載のプロセッサは、請求項1から3記載のプロセッサにおいて、さらに、前記実行手段により前記定数レジスタから定数が読み出されたことを示す状態保持手段を備え、前記格納手段は、前記状態保持手段により定数が読み出されたことを示されていると、前記第1の定数を前記定数レジスタに格納する前に前記定数レジスタの内容を消去するとしたものである。

【0014】また請求項6記載のプロセッサは、前記実行手段は、請求項4または5記載のプロセッサにおいて、前記状態保持手段により定数が読み出されたことを示されていても、前記定数レジスタに格納されている定数を繰り返し読み出し、その定数の少なくとも一部をオペランドとするオペレーションを実行するとしたものである。

【0015】これによって、定数レジスタに復元格納された定数は繰り返し読み出される。さらに請求項7記載のプロセッサは、請求項1から3記載のプロセッサにおいて、前記解読手段は、さらに、前記命令中に第3の定数を伴う定数置換オペレーションが置かれていることを解読し、前記格納手段は、前記解読手段により前記定数置換オペレーションが置かれていると解読された場合に、前記定数レジスタに格納されている定数の一部を残したまま他の部分と前記第3の定数とを置換して格納するとしたものである。

【0016】これによって、定数レジスタに復元格納された定数は一部が置き換えられつつ繰り返し読み出される。ここで請求項8記載のプロセッサは、請求項7記載のプロセッサにおいて、前記置換する部分が、前記定数

10

20

30

40

50

レジスタの最下位ビットを含む部分であるとしたものである。

【0017】さらに上記目的を達成するために請求項9記載のプロセッサは、複数の命令を格納する記憶装置から命令を読み出す命令読み出し手段と、定数を格納するための定数レジスタと、前記命令中に第1の定数が置かれていることと、前記命令中に第2の定数が置かれていることを解読する解読手段と、前記解読手段により前記第1の定数が置かれていると解読された場合に、前記定数レジスタに定数が格納されていないと判断すると前記第1の定数を所定位置に格納し、前記定数レジスタに定数が既に格納されていると判断すると前記定数レジスタに既に格納されている定数を残したまま前記第1の定数を新たに前記定数レジスタに格納する格納手段と、前記実行手段により前記定数レジスタから定数が読み出されたことを示す状態保持手段と、前記解読手段により前記第2の定数が置かれていると解読された場合であって、前記状態保持手段により定数が読み出されていないことを示されている場合は、前記定数レジスタに格納されている定数を読み出し、その読み出された定数と前記第2の定数とを連結した定数をオペランドとするオペレーションを実行し、前記状態保持手段により定数が読み出されたことを示されている場合は、前記第2の定数のみをオペランドとするオペレーションを実行する実行手段とを備えることを特徴とする。

【0018】これによって、複数の命令に跨って分割配置されていた第1の定数の断片が定数レジスタに格納されて蓄積され、元の定数に復元される。第1の定数を初めて読み出す時には第2の定数も併せて使用される。さらに上記した請求項1から9記載の各プロセッサは、命令のフォーマットを指定するフォーマットコードが置かれるフォーマットフィールドと、並列実行させるオペレーションを指定する複数のオペレーションフィールドとを含む命令を解読し実行するVLIW方式のプロセッサであって、前記解読手段は、前記フォーマットコードを参照することにより、少なくとも1つの前記オペレーションフィールドに定数が置かれていることを解読することができる。

【0019】

【発明の実施の形態】以下、本発明に係るプロセッサの実施の形態について、図面を用いて詳細に説明する。なお、本明細書では、「命令」とは本プロセッサが同時並列に解読し実行するコード全体を意味し、「オペレーション」とは本プロセッサが並列に実行できる数値演算、論理演算、転送、分岐等の処理単位又はその処理単位を指定するためのコードを意味する。

(命令フォーマット) まず、本プロセッサが解読実行する命令の構造について説明する。

【0020】本プロセッサは、VLIWアーキテクチャを採るプロセッサ(以下、「VLIWプロセッサ」とい

う。)であり、32ビット固定長の命令を解読実行する。図1(a)は、本プロセッサが実行する命令50のフィールド構成を示す図である。図1(b)~図1(d)は16種類の命令フォーマットを示す図であり、そのうち、図1(b)は3オペレーション、図1(c)は2オペレーション、図1(d)は1オペレーションを同時に指定できる命令フォーマットである。

【0021】この命令50は、32ビット固定長であり、4ビットずつに区切られた8個のフィールド(上位よりP0、0フィールド51、P1、0フィールド52、…、P3、2フィールド58)からなる。なお、P2、0フィールド53~P2、2フィールド55のグループをまとめて第1演算フィールド59と呼び、P3、0フィールド56~P3、2フィールド58のグループをまとめて第2演算フィールド60と呼ぶ。

【0022】図1(b)~図1(d)において、“const”は定数であり、これが用いられるオペレーションの種類によっては即値、絶対番地、ディスプレースメント等の数値定数や文字定数を意味する。“op”はオペレーションの種類を指定するオペコードを、“Rs”はソースオペランドとなるレジスタを、“Rd”はデスティネーションオペランドとなるレジスタを、“cc”は本プロセッサが備える専用の32ビットレジスタ(図3に示される定数レジスタ36)の格納値を分岐先の絶対番地又は相対番地(ディスプレースメント)とする分岐オペレーションを指定するオペコードを意味する。

【0023】各命令フォーマット毎の実際の動作については、後で詳細な説明を加える。図2は、図1で用いられている3種類のオペコード“cc”、“opl”及び“op2”それぞれによって指定される具体的なオペレーションを説明する図である。4ビットのオペコード“cc”は、16種類の分岐オペレーションの中の一つを指定する。1つの分岐オペレーションは、分岐条件と分岐形式によって特定される。分岐条件には、等しい(“eq”)、等しくない(“neq”)、より大きい(“gt”)等がある。分岐形式には、上記定数レジスタ36の格納値を分岐先の絶対番地として分岐する形式(ニーモニック表示において“i”が添付されていないもの)と相対番地として分岐する形式(ニーモニック表示において“i”が添付されているもの)とがある。例えば、オペコード“eq”は、直前の比較結果が等しい場合に絶対番地指定による分岐を行なうオペレーションを意味し、オペコード“eqi”は、直前の比較結果が等しい場合に相対番地指定による分岐を行なうオペレーションを意味する。

【0024】4ビットのオペコード“opl”は、“add”(加算)、“sub”(減算)、“mul”(乗算)、“and”(論理積)、“or”(論理和)等の算術論理演算に属するオペレーションの一つを指定する場合と、“mov”(ワード(32ビット)データの転

10

20

30

40

50

送)、“movh”(ハーフワードデータの転送)、“movb”(バイトデータの転送)等のレジスタ間転送に属するオペレーションの一つを指定する場合とがある。また、転送オペレーションの一つとして、定数レジスタの下位4ビットの内容を置換する“setl4”オペレーションが用意されている。

【0025】4ビットのオペコード“op2”は、上記オペコード“op1”と同様の算術論理演算及びレジスタ間転送に加えて、“ld”(メモリからレジスタへの1ワードデータのロード)、“st”(レジスタからメモリへのワードデータのストア)等のレジスタ・メモリ間転送に属するオペレーションの一つを指定する場合がある。

【0026】第1演算フィールド59には、本プロセッサと外部(メモリ)とのデータの転送を伴わないオペレーション(算術論理演算、レジスタ間転送)を指定するためのオペコードとオペランド(ソース及びデスティネーション)との組又は定数が置かれる。第2演算フィールド60には、上記第1演算フィールド59の場合に加えて、本プロセッサと外部(メモリ)とのデータの転送を伴うオペレーション(レジスタ・メモリ間転送)を指定するためのオペコードとオペランドとの組が置かれることもある。

【0027】なお、以上のようなオペレーションの種類各フィールドへの割り当ては、ノイマン型の本プロセッサにおいては2つ以上の分岐オペレーションを同時に実行する必要がないこと、本プロセッサと外部(メモリ)とのオペランドの入出力ポート(図3におけるオペランドアクセス部40)を1つに限定していること等に基づく。

(プロセッサのハードウェア構成)次に、本プロセッサのハードウェア構成を説明する。

【0028】図3は、本発明に係るプロセッサのハードウェア構成を示すブロック図である。本プロセッサは、上述したように、最大3つのオペレーションを並列実行するVLIWプロセッサであり、大きく分けて、命令レジスタ10、解読部20及び実行部30から構成される。

【0029】命令レジスタ10は、命令フェッチ部39から送られてきた1個の命令を保持する32ビットのレジスタである。解読部20は、命令レジスタ10に保持された命令を解読し、その解読結果に応じた制御線を実行部30に出力するものであり、大きく分けて、フォーマットデコーダ21と命令デコーダ22とからなる。

【0030】命令デコーダ22はさらに、P1.0フィールド12に保持されたオペコード“cc”を解読しその結果に基づいてPC部33を制御する分岐デコーダ23と、P2.0フィールド13に保持されたオペコードを解読しその結果に基づいて第1演算部37を制御する第1演算デコーダ24と、P3.0フィールド16に保

持されたオペコードを解読しその結果に基づいて第2演算部38及びオペランドアクセス部40を制御する第2演算デコーダ25とからなる。

【0031】フォーマットデコーダ21は、P0.0フィールド11に保持された4ビットのフォーマットコードをデコードすることによって命令レジスタ10に保持された命令のフォーマットが図1(b)～図1(d)に示された16種類のうちのいずれであるかを特定し、その結果に応じて分岐デコーダ23、第1演算デコーダ24及び第2演算デコーダ25による解読動作を許可又は禁止したり、実行部30の定数レジスタ制御部32を動作させたりする。

【0032】なお、上記デコーダ21、23～25は、基本的には1サイクルに1つのオペレーションを解読し、実行部30に制御信号を与える。また、命令レジスタ10と実行部30を接続する26ビットの定数信号線26は、命令レジスタ10に置かれた定数やオペランドを実行部30に転送するためのバスである。実行部30は、解読部20での解読結果に基づいて、最大3つのオペレーションを並列実行する回路ユニットであり、実行制御部31、PC部33、レジスタ群34、第1演算部37、第2演算部38、命令フェッチ部39及びオペランドアクセス部40からなる。なお、この実行部30のうち定数レジスタ制御部32、PC部33及び定数レジスタ36については、別の図面においてさらに詳細な構成を示している。

【0033】実行制御部31は、解読部20での実行結果に基づいて実行部30の各構成要素33～40を制御する制御回路や配線の総称であり、通常のプロセッサが備える構成要素(タイミング制御、動作許可禁止制御、ステータス管理、割り込み制御等の回路)の他に本プロセッサに特有の定数レジスタ制御部32を有する。定数レジスタ制御部32は、フォーマットデコーダ21からの指示に基づいて命令レジスタ10に保持された4ビット又は16ビットの定数(const)を定数レジスタ36に格納する制御を行なう。

【0034】PC(プログラムカウンタ)部33は、分岐デコーダ23による制御の下で、次に解読実行すべき命令が置かれている図示されていない外部メモリ上のアドレスを命令フェッチ部39に出力する。命令フェッチ部39は、32ビットのIA(インストラクションアドレス)バス及び32ビットのID(インストラクションデータ)バスを通じて図示されていない外部メモリから命令ブロックをフェッチし、内部の命令キャッシュに保持すると共に、PC部33から出力されたアドレスに相当する命令を命令レジスタ10に供給する。

【0035】レジスタ群34は、16個の32ビット汎用レジスタ35と1個の32ビット定数レジスタ36から構成される。これら17個のレジスタ35、36に格納された値は、第1演算デコーダ24及び第2演算デ

10

20

30

40

50

ータ25での解読結果に基づいて、第1演算部37及び第2演算部38に転送され、ここで演算が施され、又は、ここを単に通過した後に、レジスタ群34又はオペランドアクセス部40に送られる。なお、定数レジスタ36に格納された値は、第1演算部37及び第2演算部38での演算に用いられる他に、PC部33にも転送され、ここで分岐先となる有効アドレスを生成するために用いられる。

【0036】第1演算部37は、2個の32ビットデータに対して算術論理演算を行なうALUと乗算を行う乗算器とを内部に有し、第1演算デコーダ24による制御の下で2種類のオペレーション（算術論理演算とレジスタ間転送）を実行する。第2演算部38も、第1演算部37と同様に、2個の32ビットデータに対して算術論理演算を行なうALUと乗算を行う乗算器とを内部に有し、第2演算デコーダ25による制御の下で2種類のオペレーション（算術論理演算とレジスタ間転送）を実行する。

【0037】オペランドアクセス部40は、第2演算デコーダによる制御の下でレジスタ群34と図示されていない外部メモリとの間でオペランドの転送を行なう回路であり、そのオペランドやオペランドアドレスを保持するバッファを内部に有する。具体的には、例えば、命令レジスタ10のP3、1フィールド16にオペコード“ld”が置かれていた場合には、外部メモリに置かれていた1ワードのデータがオペランドアクセス部40を経てレジスタ群34のいずれかのレジスタにロードされ、また、オペコード“st”が置かれていた場合には、レジスタ群34のいずれかのレジスタの格納値が外部メモリにストアされる。

【0038】上記PC部33、レジスタ群34、第1演算部37、第2演算部38及びオペランドアクセス部40は、図示されるように、内部バス（L1バス、R1バス、L2バス、R2バス、D1バス、D2バス）で接続されている。なお、L1バス及びR1バスはそれぞれ第1演算部37の2つの入力ポートに、L2バス及びR2バスはそれぞれ第2演算部38の2つの入力ポートに、D1バス及びD2バスはそれぞれ第1演算部37及び第2演算部38の出力ポートに接続されている。

（定数レジスタ36及びその周辺回路の詳細な構成）次に、定数レジスタ36及びその周辺回路について詳細に説明する。

【0039】図4は、定数レジスタ36及びその周辺回路の詳細な構成と接続関係を示すブロック図である。なお、図中の固定値（“0”）27は、定数“0”を示す4本の信号線の固定的な配線を意味する。定数レジスタ制御部32は、5個の3入力セクタ32a～32eと3個の4入力セクタ32f～32hとからなり、定数レジスタ36は、8個の4ビット幅レジスタ36a～36hからなる。なお、各入出力データは並列4ビットで

ある。

【0040】読み出しフラグ記憶部28は、定数レジスタ36をゼロクリアすべきかどうかを調べるために用いられる。図5は、読み出しフラグ記憶部28の値の変化を示す状態遷移図である。読み出しフラグ記憶部28の値は、定数レジスタ36の格納値が読み出され、かつ定数レジスタ36への定数の格納が行なわれなかった場合に“1”にセットされる。読み出しフラグ記憶部28の値が“1”の状態、定数を格納するオペレーションが指定された場合、定数を設定する前に定数レジスタ制御部32の制御により定数レジスタ36の内容はゼロクリアされる。その後、読み出しフラグ記憶部28の値は“0”にセットされる。

【0041】定数レジスタ制御部32は、フォーマットデコーダ21及び命令デコーダ22からの制御信号に従って上記8個の入力セクタ32a～32hを制御することで、以下に示す8通りの格納方法のいずれかの方法により、命令レジスタ10に保持された定数を定数レジスタ36に格納させる。図6(a)～図6(h)は、その8通りの格納方法を説明する図である。

【0042】図6(a)は、フォーマットデコーダ21によってP0、0フィールド11に保持された値が“0”又は“1”であると解読され、読み出しフラグ記憶部28の値が“0”である場合の格納方法を示す。これは、P1、0フィールド12に置かれた4ビットの定数のみを定数レジスタ36に格納する場合に相当する。具体的には、定数レジスタ36に保持されたデータを4ビット単位で上位にシフトさせると同時に、命令レジスタ10のP1、0フィールド12に保持された4ビットの定数を定数レジスタ36の最下位の4ビットレジスタ36hに格納する。

【0043】図6(b)は、フォーマットデコーダ21によってP0、0フィールド11に保持された値が“0”又は“1”であると解読され、読み出しフラグ記憶部28の値が“1”である場合の格納方法を示す。これは、定数レジスタ36の内容をゼロクリアしてから、P1、0フィールド12に置かれた4ビットの定数のみを定数レジスタ36に格納する場合に相当する。

【0044】具体的には、命令レジスタ10のP1、0フィールド12に保持された4ビットの定数を定数レジスタ36の最下位の4ビットレジスタ36hに格納すると同時に、定数レジスタ36の残りのビット36a～36gに“0”を格納する。図6(c)は、フォーマットデコーダ21によってP0、0フィールド11に保持された値が“4”であると解読され、読み出しフラグ記憶部28の値が“0”である場合の格納方法を示す。これは、P1、0フィールド12～P2、2フィールド15に置かれた16ビットの定数を定数レジスタ36に格納する場合に相当する。

【0045】具体的には、定数レジスタ36の下位16

ビット36e~36hに保持されたデータを上位16ビット36a~36dにシフトさせると同時に、命令レジスタ10のP1. 0フィールド12~P2. 2フィールド15に保持された16ビットの定数を定数レジスタ36の下位16ビット36e~36hに格納する。図6(d)は、フォーマットデコーダ21によってP0. 0フィールド11に保持された値が“4”であると解釈され、読み出しフラグ記憶部28の値が“1”である場合の格納方法を示す。これは、定数レジスタ36の内容をゼロクリアしてから、P1. 0フィールド12~P2. 2フィールド15に置かれた16ビットの定数を定数レジスタ36に格納する場合に相当する。

【0046】具体的には、命令レジスタ10のP1. 0フィールド12~P2. 2フィールド15に保持された16ビットの定数を定数レジスタ36の下位16ビット36e~36hに格納すると同時に、残りのビット36a~36dに“0”を格納する。図6(e)は、フォーマットデコーダ21によってP0. 0フィールド11に保持された値が“5”であると解釈され、読み出しフラグ記憶部28の値が“0”である場合の格納方法を示す。これは、P1. 0フィールド12とP3. 0フィールド16~P3. 2フィールド18に置かれた16ビットの定数を定数レジスタ36に格納する場合に相当する。

【0047】具体的には、定数レジスタ36の下位16ビット36e~36hに保持されたデータを上位16ビット36a~36dにシフトさせると同時に、命令レジスタ10のP1. 0フィールド12とP3. 0フィールド16~P3. 2フィールド18に保持された16ビットの定数を定数レジスタ36の下位16ビット36e~36hに格納する。

【0048】図6(f)は、フォーマットデコーダ21によってP0. 0フィールド11に保持された値が“5”であると解釈され、読み出しフラグ記憶部28の値が“1”である場合の格納方法を示す。これは、定数レジスタ36の内容をゼロクリアしてから、P1. 0フィールド12とP3. 0フィールド16~P3. 2フィールド18に置かれた16ビットの定数を定数レジスタ36に格納する場合に相当する。

【0049】具体的には、命令レジスタ10のP1. 0フィールド12とP3. 0フィールド16~P3. 2フィールド18に保持された16ビットの定数を定数レジスタ36の下位16ビット36e~36hに格納すると同時に、残りのビット36a~36dに“0”を格納する。図6(g)は、第1演算デコーダ24でデコードした結果、オペレーションが“setl4”であると解釈された場合の格納方法を示す。これは、現在定数レジスタ36に格納されている値の下位4ビットのみを変更する場合に相当する。

【0050】具体的には、定数レジスタ36に既に格納

されているデータのシフトは行なわず、P2. 1フィールドに保持された4ビットの定数を定数レジスタ36の下位4ビット36hに格納する。図6(h)は、第2演算デコーダ25でデコードした結果、オペレーションが“setl4”であると解釈された場合の格納方法を示す。これは、現在定数レジスタ36に格納されている値の下位4ビットのみを変更する場合に相当する。

【0051】具体的には、定数レジスタ36に既に格納されているデータのシフトは行なわず、P3. 1フィールドに保持された4ビットの定数を定数レジスタ36の下位4ビット36hに格納する。以上のように、命令レジスタ10のP0. 0フィールド11の値が“0”、“1”、“4”、“5”であり、読み出しフラグ28の値が“0”の場合には、定数レジスタ36に既に格納された定数をシフトさせながら新たな定数が定数レジスタ36に格納される。そして、命令レジスタ10のP0. 0フィールド11の値が“0”、“1”、“4”、“5”であり、読み出しフラグ28の値が“1”の場合には、定数レジスタ36をゼロクリアしてから新たな定数が定数レジスタ36に格納される。また、“setl4”オペレーションが指定された場合には、定数レジスタ36の下位4ビットのみを変更し、上位ビットの格納値はそのまま残すことができる。なお、“setl4”オペレーションは、すべてのオペコード“op1”又は“op2”に配置することができ、その場合の“Rd1”又は“Rd2”の内容は無視される。

【0052】“setl4”のように、定数レジスタ36の下位ビットのみを変更するオペレーションを用意することにより、下位ビットのみが異なるいくつかの定数を使用したい場合に、定数全体を何度も設定しなおす必要がなくなり、定数レジスタ36の上位部分を使い回しすることができるようになる。図7(a)~図7(f)は、定数レジスタ36及び読み出しフラグ記憶部28の値の変化を示す図である。

【0053】図7(a)は、定数レジスタ36に格納されていた定数“0x87654321”が読み出された直後におけるそれらの内容を示し、図7(b)~図7(e)は、その後4ビットの定数“0x8”、“0x7”、“0x6”、“0x5”が順次に格納された直後におけるそれらの内容を示し、図7(f)は、図7(e)における定数“0x00008765”が読み出された直後におけるそれらの内容を示す。

【0054】このように、読み出しフラグ記憶部28によってゼロクリアの必要性の有無を管理することで、複数の命令に跨って分割配置されていた定数は、定数レジスタ36に蓄積して格納され、ゼロ拡張された元の定数として復元される。ここで、ゼロ拡張とは、ある数値の有効桁数が一定の桁数に満たない場合に、その有効桁より上位の桁全てをゼロで埋める処理をいう。

【0055】また、読み出しフラグ記憶部28を用意す

ることにより、新しい定数を格納する際にゼロクリアの必要性の有無を判定することができるようになり、定数レジスタ36の値を使用した時点で定数レジスタ36をゼロクリアしておく必要がなくなっている。これによって、以下に示すような定数レジスタ36の格納値の全体の再利用が可能となっている。

【0056】ここで、再び図1に示した各命令フォーマットについて、読み出しフラグ記憶部28の値に応じた本プロセッサの動作を説明する。なお、本プロセッサでは、初期状態として読み出しフラグ記憶部28の値は

“1”にセットされている。フォーマットコード(P0, 0フィールド11の格納値)“0”について:

(1) 読み出しフラグ記憶部28の値が“1”のとき
まず、第1演算フィールド59に指定された“op1”を実行する。ソースオペランドは“Rs1”、デスティネーションオペランドは“Rd1”である。

【0057】それと同時に、第2演算フィールド60に指定された“op2”を実行する。ソースオペランドは“Rs2”、デスティネーションオペランドは“Rd2”である。次に、P1, 0フィールド52に指定された“const”を定数レジスタ36の36hに格納し、36a~36gをゼロクリアする。そして、読み出しフラグ記憶部28の値を“0”にする。

(2) 読み出しフラグ記憶部28の値が“0”のとき
まず、第1演算フィールド59に指定された“op1”を実行する。ソースオペランドは“Rs1”、デスティネーションオペランドは“Rd1”である。

【0058】それと同時に、第2演算フィールド60に指定された“op2”を実行する。ソースオペランドは“Rs2”、デスティネーションオペランドは“Rd2”である。次に、定数レジスタ36の36b~36hを36a~36gに4ビット左シフトしてから、P1, 0フィールド52に指定された“const”を定数レジスタ36の36hに格納する。読み出しフラグ記憶部28の値は“0”のままである。

【0059】フォーマットコード“1”について:
(1) 読み出しフラグ記憶部28の値が“1”のとき
まず、第1演算フィールド59に指定された“op1”を実行する。デスティネーションオペランドは“Rd1”である。ソースオペランドはP2, 1フィールド54に指定された4ビットの定数“coust1”である。

(2) 読み出しフラグ記憶部28の値が“0”のとき

まず、第1演算フィールド59に指定された“op1”を実行する。デスティネーションオペランドは“Rd1”である。ソースオペランドは、定数レジスタ36の36b~36hとP2, 1フィールド54に指定された4ビットの定数“coust1”を連結した32ビットの定数である。そして、定数レジスタ36の内容を読み出したので、読み出しフラグ記憶部28の値を“1”にセットする。

【0061】それと同時に、第2演算フィールド60に指定された“op2”を実行する。ソースオペランドは“Rs2”、デスティネーションオペランドは“Rd2”である。次に、この時点で読みだしフラグ記憶部28の値は“1”になっているので、P1, 0フィールド52に指定された“const”を定数レジスタ36の36hに格納し、36a~36gをゼロクリアする。そして、読み出しフラグ記憶部28の値を“0”にする。

【0062】フォーマットコード“2”について:

(1) 読み出しフラグ記憶部28の値が“1”のとき
まず、第1演算フィールド59に指定された“op1”を実行する。ソースオペランドは“Rs1”、デスティネーションオペランドは“Rd1”である。それと同時に、第2演算フィールド60に指定された“op2”を実行する。ソースオペランドは“Rs2”、デスティネーションオペランドは“Rd2”である。

【0063】次に、P1, 0フィールド52に指定された“cc”オペレーションの分岐条件を満たすかどうかを判定し、満たす場合は、PC部33において定数レジスタ36の内容から分岐先アドレスを求め、PC(プログラムカウンタ)に格納する。読み出しフラグ記憶部28の値は“1”のままである。

(2) 読み出しフラグ記憶部28の値が“0”のとき
まず、第1演算フィールド59に指定された“op1”を実行する。ソースオペランドは“Rs1”、デスティネーションオペランドは“Rd1”である。

【0064】それと同時に、第2演算フィールド60に指定された“op2”を実行する。ソースオペランドは“Rs2”、デスティネーションオペランドは“Rd2”である。次に、P1, 0フィールド52に指定された“cc”オペレーションの分岐条件を満たすかどうかを判定し、満たす場合は、PC部33において定数レジスタ36の内容から分岐先アドレスを求め、PC(プログラムカウンタ)に格納する。

【0065】最後に、読み出しフラグ記憶部28の値を“1”にセットする。フォーマットコード“3”について:

(1) 読み出しフラグ記憶部28の値が“1”のとき
まず、第1演算フィールド59に指定された“op1”を実行する。デスティネーションオペランドは“Rd1”である。ソースオペランドはP2, 1フィールド54に指定された4ビットの定数“coust1”であ

る。

【0066】それと同時に、第2演算フィールド60に指定された“op2”を実行する。ソースオペランドは“Rs2”、デスティネーションオペランドは“Rd2”である。次に、P1.0フィールド52に指定された“cc”オペレーションの分岐条件を満たすかどうかを判定し、満たす場合は、PC部33において定数レジスタ36の内容から分岐先アドレスを求め、PC（プログラムカウンタ）に格納する。

【0067】読み出しフラグ記憶部28の値は“1”のままである。

(2) 読み出しフラグ記憶部28の値が“0”のときまず、第1演算フィールド59に指定された“op1”を実行する。デスティネーションオペランドは“Rd1”である。ソースオペランドはP2.1フィールド54に指定された4ビットの定数“const1”である。

【0068】それと同時に、第2演算フィールド60に指定された“op2”を実行する。ソースオペランドは“Rs2”、デスティネーションオペランドは“Rd2”である。次に、P1.0フィールド52に指定された“cc”オペレーションの分岐条件を満たすかどうかを判定し、満たす場合は、PC部33において定数レジスタ36の内容から分岐先アドレスを求め、PC（プログラムカウンタ）に格納する。

【0069】最後に、読み出しフラグ記憶部28の値を“1”にセットする。フォーマットコード“4”について：

(1) 読み出しフラグ記憶部28の値が“1”のときまず、第2演算フィールド60に指定された“op2”を実行する。ソースオペランドは“Rs2”、デスティネーションオペランドは“Rd2”である。

【0070】次に、P1.0フィールド52～P2.2フィールド55に指定された16ビットの定数“const”を定数レジスタ36の下位16ビット36e～36hに格納し、36a～36dをゼロクリアする。そして、読み出しフラグ記憶部28の値を“0”にする。

(2) 読み出しフラグ記憶部28の値が“0”のときまず、第2演算フィールド60に指定された“op2”を実行する。ソースオペランドは“Rs2”、デスティネーションオペランドは“Rd2”である。

【0071】次に、定数レジスタ36の下位16ビット36e～36hを36a～36dにシフトしてから、P1.0フィールド52～P2.2フィールド55に指定された16ビットの定数“const”を定数レジスタ36の下位16ビット36e～36hに格納する。読み出しフラグ記憶部28の値は“0”のままである。フォーマットコード“5”について：

(1) 読み出しフラグ記憶部28の値が“1”のときまず、第1演算フィールド59に指定された“op1”

を実行する。ソースオペランドは“Rs1”、デスティネーションオペランドは“Rd1”である。

【0072】次に、P1.0フィールド52とP3.0フィールド56～P3.2フィールド58に指定された16ビットの定数“const”を定数レジスタ36の下位16ビット36e～36hに格納し、36a～36dをゼロクリアする。そして、読み出しフラグ記憶部28の値を“0”にする。

(2) 読み出しフラグ記憶部28の値が“0”のときまず、第1演算フィールド59に指定された“op1”を実行する。ソースオペランドは“Rs1”、デスティネーションオペランドは“Rd1”である。

【0073】次に、定数レジスタ36の下位16ビット36e～36hを36a～36dにシフトしてから、P1.0フィールド52とP3.0フィールド56～P3.2フィールド58に指定された16ビットの定数“const”を定数レジスタ36の下位16ビット36e～36hに格納する。読み出しフラグ記憶部28の値は“0”のままである。

【0074】フォーマットコード“6”について：

(1) 読み出しフラグ記憶部28の値が“1”のときまず、第1演算フィールド59に指定された“op1”を実行する。デスティネーションオペランドは“Rd1”である。ソースオペランドはP1.0フィールド52に指定された4ビットの定数“const1”とP2.1フィールド54に指定された4ビットの定数“const1”を連結させた8ビットの定数である。

【0075】それと同時に、第2演算フィールド60に指定された“op2”を実行する。ソースオペランドは“Rs2”、デスティネーションオペランドは“Rd2”である。

(2) 読み出しフラグ記憶部28の値が“0”のときまず、第1演算フィールド59に指定された“op1”を実行する。デスティネーションオペランドは“Rd1”である。ソースオペランドは、定数レジスタ36の下位24ビット36c～36hとP1.0フィールド52に指定された4ビットの定数“const1”とP2.1フィールド54に指定された4ビットの定数“const1”を連結した32ビットの定数である。そして、定数レジスタ36の内容を読み出したので、読み出しフラグ記憶部28の値を“1”にセットする。

【0076】それと同時に、第2演算フィールド60に指定された“op2”を実行する。ソースオペランドは“Rs2”、デスティネーションオペランドは“Rd2”である。フォーマットコード“7”について：

(1) 読み出しフラグ記憶部28の値が“1”のときまず、第1演算フィールド59に指定された“op1”を実行する。ソースオペランドは“Rs1”、デスティネーションオペランドは“Rd1”である。

【0077】それと同時に、第2演算フィールド60に

指定された“op2”を実行する。デスティネーションオペランドは“Rd2”である。ソースオペランドはP1. 0フィールド52に指定された4ビットの定数“coust1”とP3. 1フィールド57に指定された4ビットの定数“coust1”を連結させた8ビットの定数である。

(2) 読み出しフラグ記憶部28の値が“0”のとき
まず、第1演算フィールド59に指定された“op1”を実行する。ソースオペランドは“Rs1”、デスティネーションオペランドは“Rd1”である。

【0078】それと同時に、第2演算フィールド60に指定された“op2”を実行する。デスティネーションオペランドは“Rd2”である。ソースオペランドは、定数レジスタ36の下位24ビット36c~36hとP1. 0フィールド52に指定された4ビットの定数“coust1”とP3. 1フィールド57に指定された4ビットの定数“coust1”を連結した32ビットの定数である。そして、定数レジスタ36の内容を読み出したので、読み出しフラグ記憶部28の値を“1”にセットする。

【0079】フォーマットコード“8”について：

(1) 読み出しフラグ記憶部28の値が“1”のとき
まず、第1演算フィールド59に指定された“op1”を実行する。デスティネーションオペランドは“Rd1”である。ソースオペランドはP1. 0フィールド52に指定された4ビットの定数“coust1”とP2. 1フィールド54に指定された4ビットの定数“coust1”を連結させた8ビットの定数である。

【0080】それと同時に、第2演算フィールド60に指定された“op2”を実行する。デスティネーションオペランドは“Rd2”である。ソースオペランドはP3. 1フィールド57に指定された4ビットの定数“coust2”である。

(2) 読み出しフラグ記憶部28の値が“0”のとき
まず、第1演算フィールド59に指定された“op1”を実行する。デスティネーションオペランドは“Rd1”である。ソースオペランドは、定数レジスタ36の下位24ビットとP1. 0フィールド52に指定された4ビットの定数“coust1”とP2. 1フィールド54に指定された4ビットの定数“const1”を連結した32ビットの定数である。そして、定数レジスタ36の内容を読み出したので、読み出しフラグ記憶部28の値を“1”にセットする。

【0081】それと同時に、第2演算フィールド60に指定された“op2”を実行する。デスティネーションオペランドは“Rd2”である。ソースオペランドはP3. 1フィールド57に指定された4ビットの定数“coust2”である。フォーマットコード“9”について：

(1) 読み出しフラグ記憶部28の値が“1”のとき

まず、第1演算フィールド59に指定された“op1”を実行する。デスティネーションオペランドは“Rd1”である。ソースオペランドはP2. 1フィールド54に指定された4ビットの定数“coust1”である。

【0082】それと同時に、第2演算フィールド60に指定された“op2”を実行する。デスティネーションオペランドは“Rd2”である。ソースオペランドはP1. 0フィールド52に指定された4ビットの定数“coust2”とP3. 1フィールド57に指定された4ビットの定数“coust2”を連結させた8ビットの定数である。

(2) 読み出しフラグ記憶部28の値が“0”のとき
まず、第1演算フィールド59に指定された“op1”を実行する。デスティネーションオペランドは“Rd1”である。ソースオペランドは、定数レジスタ36の下位28ビット36b~36hとP1. 0フィールド52に指定された4ビットの定数“coust1”とを連結した32ビットの定数である。そして、定数レジスタ36の内容を読み出したので、読み出しフラグ記憶部28の値を“1”にセットする。

【0083】それと同時に、第2演算フィールド60に指定された“op2”を実行する。デスティネーションオペランドは“Rd2”である。ソースオペランドはP1. 0フィールド52に指定された4ビットの定数“coust2”とP3. 1フィールド57に指定された4ビットの定数“coust2”を連結させた8ビットの定数である。

【0084】フォーマットコード“A”について：

(1) 読み出しフラグ記憶部28の値が“1”のとき
まず、第2演算フィールド60に指定された“op2”を実行する。デスティネーションオペランドは“Rd1”である。ソースオペランドは、P2. 0フィールド53~P2. 2フィールド55に指定された12ビットの定数“coust2”とP3. 1フィールド57に指定された4ビットの定数“coust2”を連結させた16ビットの定数である。

【0085】次に、P1. 0フィールド52に指定された“cc”オペレーションの分岐条件を満たすかどうかを判定し、満たす場合は、PC部33において定数レジスタ36の内容から分岐先アドレスを求め、PC（プログラムカウンタ）に格納する。読み出しフラグ記憶部28の値は“1”のままである。

(2) 読み出しフラグ記憶部28の値が“0”のとき
まず、第2演算フィールド60に指定された“op2”を実行する。デスティネーションオペランドは“Rd1”である。ソースオペランドは、P2. 0フィールド53~P2. 2フィールド55に指定された12ビットの定数“coust2”とP3. 1フィールド57に指定された4ビットの定数“coust2”を連結させた

16ビットの定数である。

【0086】次に、P1. 0フィールド52に指定された“cc”オペレーションの分岐条件を満たすかどうかを判定し、満たす場合は、PC部33において定数レジスタ36の内容から分岐先アドレスを求め、PC（プログラムカウンタ）に格納する。最後に、読み出しフラグ記憶部28の値を“1”にセットする。

【0087】フォーマットコード“B”について：1オペレーションのみを実行する命令であり、読み出しフラグ記憶部28の値にかかわらず、定数レジスタ36の内容の参照は行なわれない。オペランドは、P1. 0フィールド52～P2. 2フィールド55に指定された16ビットの定数“coust2”とP3. 1フィールド57～P3. 2フィールド58に指定された8ビットの定数“coust2”を連結させた24ビットの定数である。

【0088】フォーマットコード“C”～フォーマットコード“F”については、“reserved”として、拡張命令などのために取ってある。本プロセッサでは、1個の命令に最大3つのオペレーションを指定することができるが、その場合には、図1（b）に示された3オペレーション用の命令フォーマットから判るように、それら3つのオペレーションの種類は次のいずれかの組み合わせになる。

（1）4ビットの定数を定数レジスタ36にセットするオペレーションと2個の汎用オペレーション（フォーマットコードが“0”、“1”の場合）

（2）定数レジスタ36にセットされた値を絶対番地又は相対番地として分岐するオペレーションと2個の汎用オペレーション（フォーマットコードが“2”、“3”の場合）

このように、本プロセッサの命令は、わずか32ビット長でありながら最大3つのオペレーションを同時に指定することができるコード効率の高いフィールド構成を有している。また、定数レジスタ36の格納値を読み出すと同時に、読み出した定数とオペランドフィールドに指定された定数を連結した定数をオペランドとしてオペレーションを実行することができることが特徴である。

（PC部33の詳細な構成）次に、PC部33の詳細な構成を説明する。

【0089】図8は、PC部33の詳細な構成を示すブロック図である。PC部33は、定数“4”を示す固定的な配線である固定値（“4”）33a、2入力セクタ33b、加算器33c、次に解読実行すべき命令のアドレスを保持するPC33d及び4入力セクタ33eから構成される。このPC部33では、解読部20からの制御信号に従ってセクタ33b、33eが動作することにより、以下の3通りの値のいずれかが有効アドレスとしてセクタ33eから命令フェッチ部39に出力される。

（1）PC33dの内容に“4”を加算した値

これは、分岐しないで順次に行なう場合、即ち、解読実行された命令に分岐オペレーションが指定されていない場合に相当する。なお、“4”を加算するのは、1つの命令の長さが4バイト（32ビット）であることによる。

（2）PC33dの内容に定数レジスタ36の内容を加算した値

これは、定数レジスタ36の内容を相対番地として分岐する場合、例えば、P1. 0フィールド12によって相対番地による分岐が指定されていると分岐デコーダ23が解読した場合が該当する。

（3）定数レジスタ36の内容

これは、定数レジスタ36の内容を絶対番地として分岐する場合、例えば、P1. 0フィールド12によって絶対番地による分岐が指定されていると分岐デコーダ23が解読した場合が該当する。

【0090】以上のように、このPC部33は、専用の加算器33cを備え、定数レジスタ36に保持された値を直接用いる構成となっているので、第1演算部37や第2演算部38での演算とは独立並行して、定数レジスタ36の格納値を絶対番地又は相対番地として分岐する実行制御を行なうことができる。

（プロセッサの動作）次に、具体的な命令を解読実行した場合の本プロセッサの動作について説明する。

【0091】図9は、24ビットの定数を扱う処理の一例を示すフローチャートである。本図には、レジスタR0とR1との格納値の差を求め（ステップS100）、その結果に24ビットの定数“0x876543”（以下、“0x”を付した数は16進数を表す）を加え（ステップS101、S102）、最後にレジスタR2の格納値をレジスタR1に転送しておく（ステップS103）という処理が示されている。

【0092】図10は、図9に示された処理内容を本プロセッサに行なわせるプログラムの例を示す図である。このプログラムは、2個の命令71、72から構成されている。1行が1個の命令に相当し、各命令の内容は各フィールドに置かれたニーモニックで表現されている。

なお、定数は全て16進数で表現されている。また、“fmt n（n=0～F）”はフォーマットコード“n”を示し、“Rn（n=0～15）”はレジスタ群34の中の一つのレジスタを示す。

【0093】図10を用いて、各命令71、72ごとの本プロセッサの動作を説明する。まず、初期状態として、読み出しフラグ記憶部28の値を“1”にする。

（命令71）命令71が命令レジスタ10にロードされると、フォーマットデコーダ21は、命令レジスタ10のP0. 0フィールド11の値（“fmt 4”）から、この命令はフォーマットコードが“4”の2オペレーション命令であると判断し、以下の2つのオペレーション

が並列実行されるように実行部 30 を制御する。

(1) 第 1 のオペレーション

定数レジスタ制御部 32 は、内部の 8 個のセクタ 32 a ~ 32 h を制御することで、図 6 (c) に示された格納方法により、P1. 0 フィールド 12 ~ P2. 2 フィールド 15 に保持された 16 ビットの定数 (0x8765) を定数レジスタ 36 の下位 16 ビットに格納する。定数レジスタ 36 の上位 16 ビットはゼロクリアされる。そして、読み出しフラグ記憶部 28 の値を "0" にする。

(2) 第 2 のオペレーション

第 2 演算部 38 は、汎用レジスタ R0 の内容と汎用レジスタ R1 の内容を入力とし、R0 の内容から R1 の内容を減算した後に、その結果を再び汎用レジスタ R0 に格納する。

(命令 72) 命令 72 が命令レジスタ 10 にロードされると、フォーマットデコーダ 21 は、命令レジスタ 10 の P0. 0 フィールド 11 の値 ("f m t 6") から、この命令はフォーマットコードが "6" の 2 オペレーション命令であると判断し、以下の 2 つのオペレーションが並列実行されるように実行部 30 を制御する。

(1) 第 1 のオペレーション

P2. 0 フィールド 13 のオペレーションは、定数をオペランドにとるオペレーションであり、読み出しフラグ記憶部 28 の値が "0" なので、定数レジスタ 36 の下位 24 ビット (0x008765) と P1. 0 フィールド 12 に指定された 4 ビットの定数 (0x4) と P2. 1 フィールド 14 に指定された 4 ビットの定数 (0x3) を連結した値 (0x00876543) と汎用レジスタ R0 の内容とで加算を行なった後に、その結果を再び汎用レジスタ R0 に格納する。

【0094】そして、読み出しフラグ記憶部 28 の値を "1" にセットする。

(2) 第 2 のオペレーション

第 2 演算部 38 は、汎用レジスタ R2 の値を入力とし、そのまま通過させて、汎用レジスタ R1 に格納する。以上のようにして、暗黙的に定数レジスタ 36 の格納値を利用し、定数レジスタ 36 とオペレーションで指定された定数を連結した定数をオペランドとして実行することにより、2 個の命令によって 24 ビットの定数を扱うオペレーションを実行することができる。

(定数レジスタの読み出しと同時に使用することができる場合との比較) 次に、本発明に係るプロセッサのように定数レジスタ 36 の格納値を暗黙のうちに使用するのではなく、従来の技術の項に示した特願平 9-159049 のプロセッサのように、明示的に定数レジスタを指定してオペレーションを行ない、定数レジスタ 36 へ定数を格納するオペレーションと定数レジスタ 36 の格納値を利用するオペレーションとが分離しているプロセッサにおいて、上記図 10 に示されたプログラムと同一

の処理を行なったときの処理方法について説明し、本発明に係るプロセッサと比較する。

【0095】図 11 は、上記従来のプロセッサにおいて、図 10 と同一内容の処理を行なうプログラムの例を示す。ただし、定数レジスタ名は R15 であるとする。図 11 と図 10 を比較して判るように、上記従来のプロセッサ用のプログラムは、本発明に係るプロセッサ用のものよりも 1 個の命令だけ多くなっている。図 11 においては、1 つのオペレーションでは、定数レジスタへの値の設定もしくは定数レジスタの格納値の利用のいずれか一方しか行なうことができないため、24 ビットの定数を設定するのに必要な 2 個の命令と、その定数を利用する命令とを合わせた 3 個の命令が必要になっている。その結果、無動作オペレーション "nop" を挿入せざるを得なくなっている。

【0096】以上のように、本発明に係るプロセッサによれば、定数レジスタに蓄積された定数とオペレーションで指定された定数からオペランドとなる定数を生成すると同時にその定数を用いてオペレーションを実行することができる。これにより、実行サイクル数を減少させることができる。

(定数レジスタ全体の再読み出し) 次に、分岐オペレーションを含むプログラムの例を図 12 に示す。

【0097】このプログラムは、3 個の命令 76 ~ 78 から構成されている。図 12 を見て判るように、本プログラムでは、命令 76 において図 6 (c) の格納方法にて 16 ビットの定数 "0x1234" を定数レジスタ 36 に設定し、命令 77 の "eq i" オペレーションにて定数レジスタ 36 の内容を参照し、その値 (0x1234) を相対番地 (ディスプレースメント) として分岐オペレーションを実行している。この時点で読み出しフラグ記憶部 28 の値は "1" にセットされる。そして、命令 78 の "gt i" オペレーションにて再び定数レジスタ 36 の内容を参照し、その値 (0x1234) を相対番地として分岐オペレーションを実行している。なお、分岐条件の判定には、直前の "cmp" オペレーションの結果が反映される。

【0098】このように、読み出しフラグ記憶部 28 の値が "1" にセットされても、次に定数レジスタ 36 に定数を格納するオペレーションが実行されるまでは、定数レジスタ 36 の内容は変更されないで、定数レジスタ 36 の格納値全体を複数回読み出して利用することができる。これによって、同一の定数を定数レジスタ 36 に何度も設定し直す必要がなくなり、実行サイクル数・命令コードサイズを低減することができる。

(定数レジスタの一部を使い回し) 次に、図 6 の定数レジスタ 36 への定数の格納方法の説明で現れた "set 14" オペレーションを含むプログラムの例を図 13 に示す。

【0099】このプログラムは、3 個の命令 79 ~ 81

10

20

30

40

50

から構成されている。図13を見て判るように、本プログラムでは、命令79において図6(c)の格納方法にて16ビットの定数“0x3210”を定数レジスタ36に設定し、命令80の“eqi”オペレーションにて定数レジスタ36の内容を参照し、その値(0x3210)を相対番地(ディスプレースメント)として分岐オペレーションを実行している。また、命令80の第二演算フィールド60の“setl4”オペレーションによって図6(h)の格納方法にて4ビットの定数“0xc”を定数レジスタ36の下位4ビットに格納する。これにより定数レジスタ36の格納値は“0x321c”になる。次に、命令81の“gti”オペレーションにて定数レジスタ36の内容を参照し、その値(0x321c)を相対番地として分岐オペレーションを実行している。なお、分岐条件の判定には、直前の“cmp”オペレーションの結果が反映される。

【0100】このように、“setl4”のような定数レジスタ36の一部のみを変更するオペレーションを用意することにより、一部分のみ異なる定数の設定は、異なる部分のみ設定し直せばよいことになる。これによって、一部分のみ異なる定数を定数レジスタ36に何度も一から設定し直す必要がなくなり、実行サイクル数・命令コードサイズを低減することができる。

(定数レジスタ36の周辺回路の変形例)次に、図4に示された定数レジスタ36の周辺回路についての変形例を示す。上記実施例では、定数レジスタ36の下位4ビットのみを変更するオペレーションを用意し、上位ビットの格納値を使い回しできることを示したが、下位4ビットだけでなく、任意のビットについて上記のような格納値の一部の再利用が可能である。その一例として、図14に、定数レジスタ36の周辺回路の変形例のブロック図を示す。

【0101】図4の定数レジスタ制御部32は定数レジスタ36をシフトレジスタとして機能させたが、図14の定数レジスタ制御部90は定数レジスタ36を並列入力のレジスタとして機能させている点で異なる。具体的には、定数レジスタ制御部90とフォーマットデコーダ21との接続及び定数レジスタ制御部90の構成要素が図4に示されたものと異なる。

【0102】定数レジスタ制御部90は、格納桁カウンタ91と8個の8入力セレクタ90a~90hとからなる。格納桁カウンタ91は、3ビットのカウンタであり、その時点において定数レジスタ36に蓄積されている定数の有効桁数をニブル(4ビット)単位で示すものである。定数レジスタ制御部90は、命令レジスタ10に保持された定数を格納する旨の指示をフォーマットデコーダ21から受けると、その時点での格納桁カウンタ91の値を参照することで定数レジスタ36の適切な位置にその定数を格納し、その後、格納桁カウンタ91を更新しておく。

【0103】ただし、図4の定数レジスタ制御部32の場合と異なり、後で実行される命令に置かれた定数が定数レジスタ36の上位桁に格納される。したがって、例えば32ビットの定数を16ビットずつ2回に分けて格納する場合には、先に下位16ビットを格納することになる。定数レジスタを図14のように構成し、定数レジスタ36a~36hの任意のレジスタの格納値のみを変更するオペレーション“set4a”、“set4b”、…、“set4h”や、定数レジスタ36の下位8ビットや上位8ビットのみを変更するオペレーション“setl8”、“setu8”などのオペレーション群を用意することにより、定数レジスタ36の下位のみに限らない一部の格納値のみを変更することが可能となり、残りの格納値を使い回しすることができるようになる。

【0104】以上、本発明に係るプロセッサについて、実施形態及び変形例に基づいて説明したが、本発明はこれら実施形態及び変形例に限られないことは勿論である。即ち、

(1) 上記実施の形態では、数値定数を扱う例が示されたが、文字定数であってもよいことは言うまでもない。複数の命令に跨って分割配置された文字定数であっても、定数レジスタ36への複数回の格納によって、桁数の長い元の文字定数が復元されるからである。

(2) また、上記実施の形態では、定数レジスタ36に定数が格納される時にゼロ拡張される構成になっていたが、格納される時に符号拡張される構成にしてもよいし、また、定数を読み出す際にゼロ拡張/符号拡張を行なう構成にしてもよい。そのためには、定数レジスタ36の周辺回路に拡張を制御する回路を追加すればよい。

【0105】ここで、符号拡張とは、ある数値の有効桁数が一定の桁数に満たない場合に、その数値の最上位ビットを符号ビットとみなし、その符号ビットと同じ論理値で有効桁より上位の桁全てを埋める処理をいう。

(3) また、上記実施の形態では、定数レジスタ36に格納されていた値の下位部分(上位部分の値を捨てたもの)の下位側に、オペレーションで指定された定数を連結し、それをオペランドとしてオペレーションを実行することを可能としたが、オペレーションで指定された定数を定数レジスタ36に格納されていた値の下位部分の上位側に連結し、それをオペランドとしてオペレーションを実行することも可能である。さらに、定数レジスタ36の上位部分の値を捨てるのではなく下位部分の値を捨て、定数レジスタ36の上位部分の下位側または上位側にオペレーションで指定された定数を連結し、それをオペランドとしてオペレーションを実行することも可能である。

(4) また、上記実施の形態では、図1(b)~図1(d)の命令フォーマットから判るように、1個の命令によって定数レジスタ36に格納させることができる定

数の桁数は4ビット及び16ビットのいずれかであったが、本発明はこの桁数に限定されるものではない。例えば、12ビットや28ビットの定数を定数レジスタ36に格納するための命令フォーマットを定義してもよい。そのためには、定数レジスタ36の周辺回路の接続関係を変更すればよい。

(5) また、上記実施の形態のプロセッサは2個の演算部37、38を備えるVLIWプロセッサであったが、本発明は、1個の演算部のみを備え1個の命令中に1個のオペレーションだけを指定する単一オペレーション命令を実行するVLIWアーキテクチャを採らないプロセッサに対しても適用することができるのは言うまでもない。

【0106】

【発明の効果】以上の説明から明らかなように、本発明のプロセッサは、複数の命令を格納する記憶装置から命令を読み出す命令読み出し手段と、定数を格納するための定数レジスタと、前記命令中に第1の定数が置かれていることと、前記命令中に第2の定数が置かれていることを解読する解読手段と、前記解読手段により前記第1の定数が置かれていると解読された場合に、前記第1の定数を前記定数レジスタに格納する格納手段と、前記解読手段により前記第2の定数が置かれていると解読された場合に、前記定数レジスタに格納されている定数を読み出し、その読み出された定数と前記第2の定数とを連結した定数をオペランドとするオペレーションを実行する実行手段とを備えることを特徴とする。

【0107】これによって、第2の定数がオペランドとして使用される際に定数レジスタに格納された第1の定数も併せて使用される。その結果、1つのオペレーションで定数の設定とその定数の使用とを同時に行なうことが可能となり、実行サイクル数を削減することができる。また本発明のプロセッサは、前記実行手段により前記定数レジスタから定数が読み出されたことを示す状態保持手段とを備え、前記格納手段は、前記状態保持手段により定数が読み出されたことを示されていると、前記定数を前記定数レジスタに格納する前に前記定数レジスタの内容を消去することを特徴とするプロセッサにおいて、前記実行手段は、前記状態保持手段により定数が読み出されたことを示されていると、前記定数レジスタに格納されている定数を繰り返し読み出し、その定数の少なくとも一部をオペランドとするオペレーションを実行するとしたものである。

【0108】これによって、定数レジスタに復元格納された定数は繰り返し読み出される。その結果、定数レジスタの格納値の全体を複数回使用することができるようになり、全く同じ定数を繰り返し設定する必要がなくなるので、実行サイクル数・命令コードサイズを低減することができる。また本発明のプロセッサは、前記解読手段は、さらに、前記命令中に第3の定数を伴う定数置換

オペレーションが置かれていることを解読し、前記格納手段は、前記解読手段により前記定数置換オペレーションが置かれていると解読された場合に、前記定数レジスタに格納されている定数の一部を残したまま他の部分と前記第3の定数とを置換して格納するとしたものである。

【0109】これによって、定数レジスタに復元格納された定数は一部が置き換えられつつ繰り返し読み出される。その結果、定数レジスタの格納値の一部を複数回使用することができるようになり、一部分のみ異なる定数を繰り返し最初から設定する必要がなくなるので、実行サイクル数・命令コードサイズを低減することができる。

【図面の簡単な説明】

【図1】(a)本発明に係るプロセッサが実行する命令のフィールド構成を示す図

(b)3オペレーションを示す図

(c)2オペレーションを示す図

(d)1オペレーションを同時に指定できる命令フォーマットを示す図

【図2】図1で用いられている3種類のオペコード“c”、“op1”及び“op2”それぞれによって指定される具体的なオペレーションを説明する図

【図3】同プロセッサのハードウェア構成を示すブロック図

【図4】同プロセッサの定数レジスタ36及びその周辺回路の詳細な構成を示すブロック図

【図5】図4に示された読み出しフラグ記憶部28の値の変化を示す遷移図

【図6】図4に示された定数レジスタ制御部32による定数の格納方法を示す図

【図7】図4に示された定数レジスタ36及び読み出しフラグ記憶部28の値の変化を示す図

【図8】同プロセッサのPC部33の詳細な構成を示すブロック図

【図9】24ビットの定数を扱う処理の一例を示すフローチャート

【図10】図9に示された処理を同プロセッサに行なわせるプログラムの例を示す図

【図11】定数を格納するオペレーションと定数レジスタ36の格納値を利用するオペレーションが分離しており、定数レジスタを明示的に指定してオペレーションを実行するプロセッサにおいて、図10と同一の処理を行なわせるプログラムの例を示す図

【図12】分岐オペレーションを含む処理を同プロセッサに行なわせるプログラムの例を示す図

【図13】定数レジスタ36の下位4ビットのみを変更するオペレーション“set14”を含む処理を同プロセッサに行なわせるプログラムの例を示す図

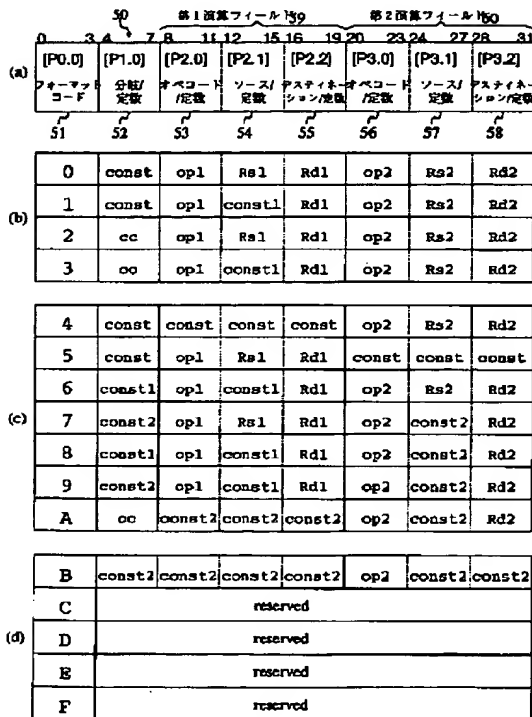
【図14】定数レジスタの変形例に係る定数レジスタ3

6の周辺回路の構成を示すブロック図

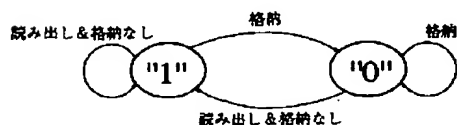
【符号の説明】

- 10 命令レジスタ
 20 解説部
 21 フォーマットデコーダ
 22 命令デコーダ
 23 分岐デコーダ
 24 第1演算デコーダ
 25 第2演算デコーダ
 28 読み出しフラグ記憶部
 30 実行部
 31 実行制御部
 32 定数レジスタ制御部
 32 a～32 h セレクタ
 33 PC部
 33 a 固定値“4”
 33 b、33 e セレクタ

【図1】



【図5】



- * 33 c 加算器
 33 d PC
 34 レジスタ群
 35 汎用レジスタR0～R15
 36 定数レジスタR16
 36 a～36 h 4ビット幅レジスタ
 37 第1演算部
 38 第2演算部
 39 命令フェッチ部
 40 オペランドアクセス部
 50 命令
 51～58 命令フィールド
 59 第1演算フィールド
 60 第2演算フィールド
 90 変形例に係る定数レジスタ制御部
 91 格納桁カウンタ

*

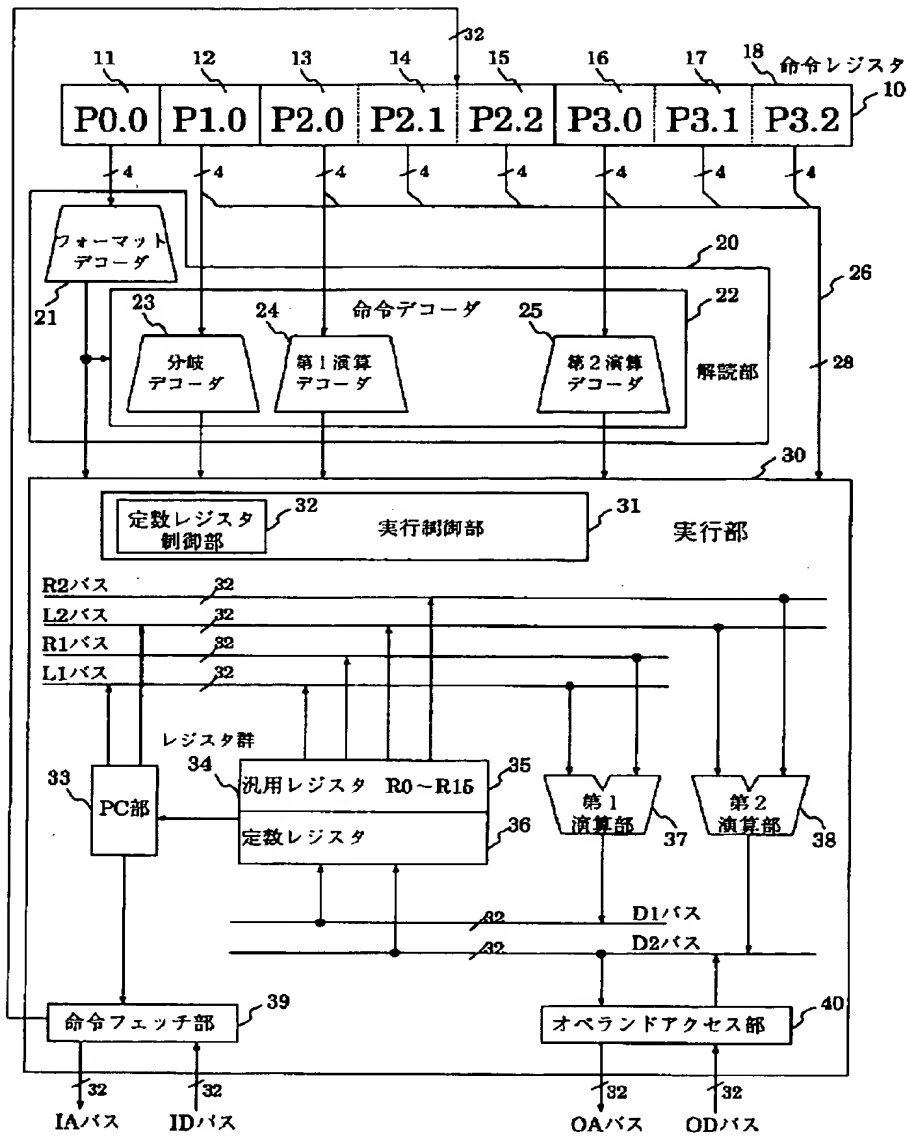
【図2】

| 記号 | オペレーション | ニーモニック表示 |
|-----|-------------|----------------------------------|
| cc | 分岐 | eq, eqi, gt, gti, jmp, jmp1, ... |
| op1 | 算術論理演算 | add, sub, mul, and, or, cmp, ... |
| | レジスタ間転送 | mov, movh, movb, setl4 |
| op2 | 算術論理演算 | add, sub, mul, and, or, cmp, ... |
| | レジスタ間転送 | mov, movh, movb, setl4 |
| | レジスタ・メモリ間転送 | ld, ldh, ldh, st, sth, stb |

【図7】

| | | | | | | | | | |
|-----|-----|-----|-----|-----|-----|-----|-----|-----|----|
| | 36a | 36b | 36c | 36d | 36e | 36f | 36g | 36h | 28 |
| (a) | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 1 |
| (b) | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 8 | 0 |
| (c) | 0 | 0 | 0 | 0 | 0 | 0 | 8 | 7 | 0 |
| (d) | 0 | 0 | 0 | 0 | 0 | 8 | 7 | 6 | 0 |
| (e) | 0 | 0 | 0 | 0 | 8 | 7 | 6 | 5 | 0 |
| (f) | 0 | 0 | 0 | 0 | 8 | 7 | 6 | 5 | 1 |

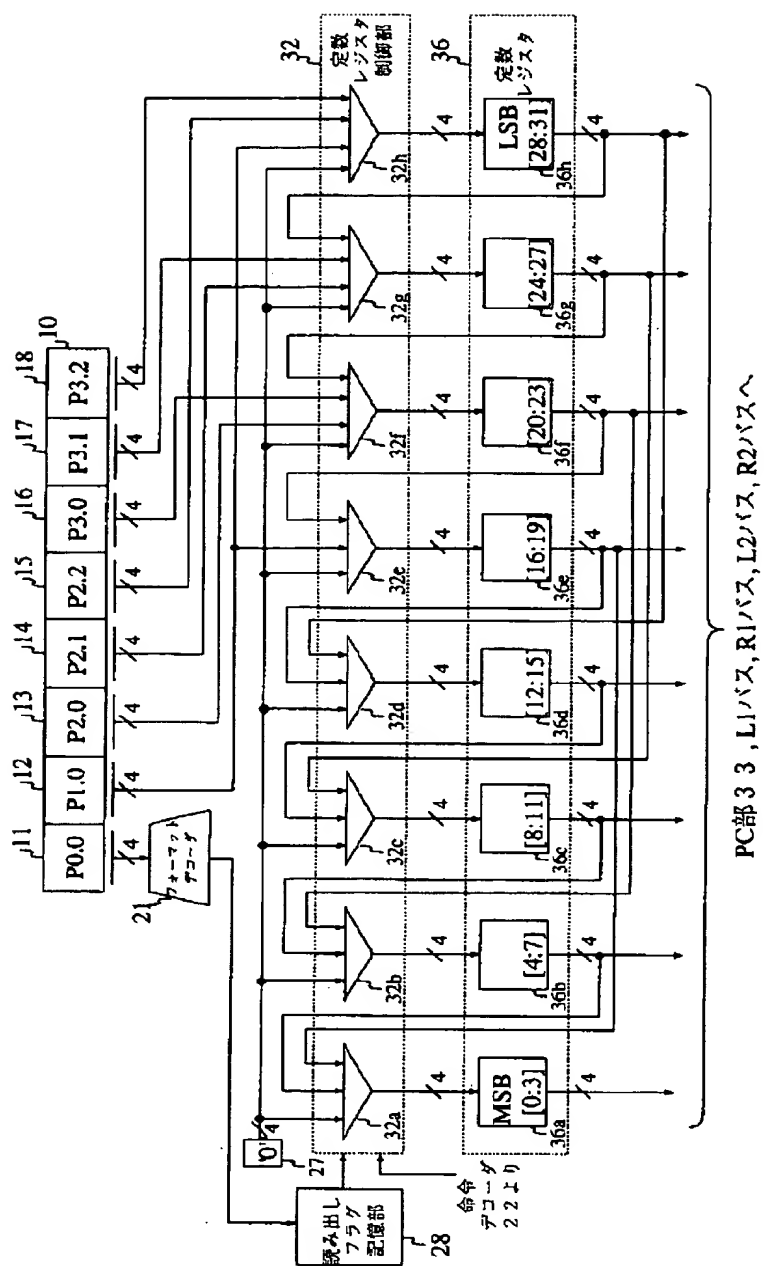
【図3】



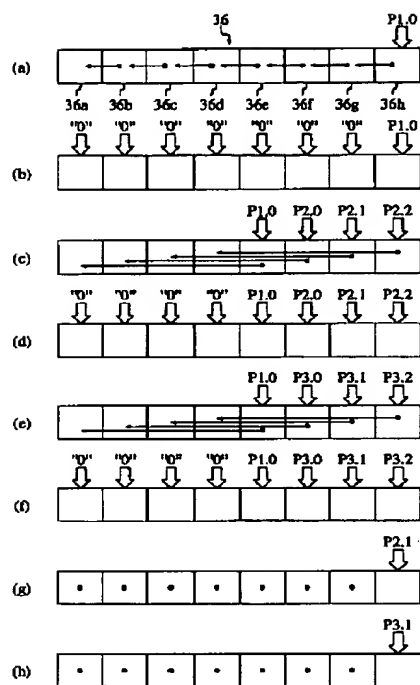
【図10】

| P0.0 | P1.0 | P2.0 | P2.1 | P2.2 | P3.0 | P3.1 | P3.2 | |
|-------|--------|------|------|------|------|------|------|----|
| fmt 4 | 0x8765 | | | | sub | R1 | R0 | 71 |
| fmt 6 | 0x4 | add | 0x3 | R0 | mov | R2 | R1 | 72 |

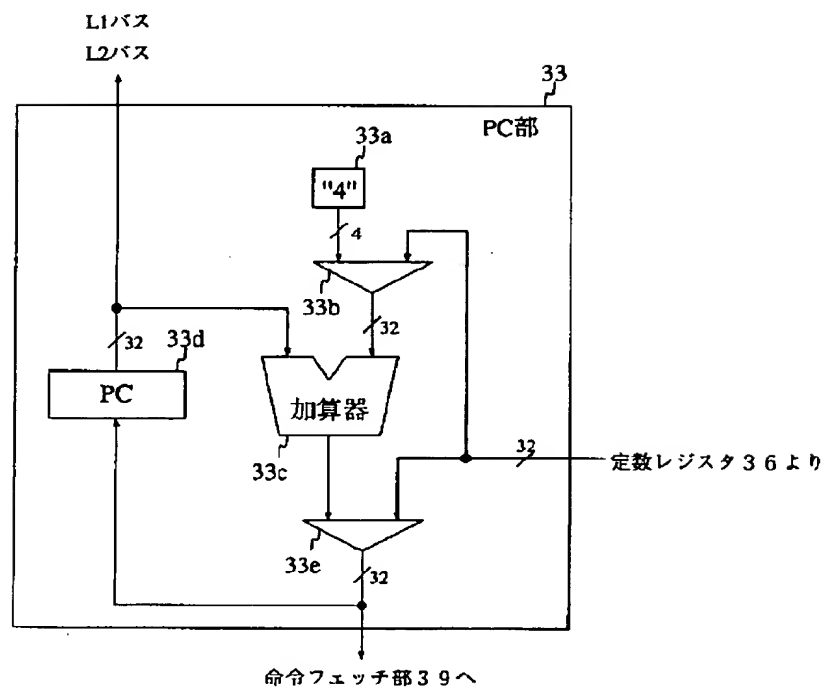
【図 4】



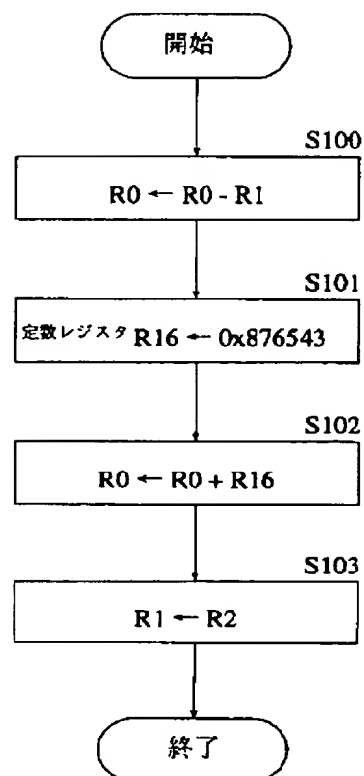
【図6】



【図8】



【図9】



【図 1 1】

| P0.0 | P1.0 | P2.0 | P2.1 | P2.2 | P3.0 | P3.1 | P3.2 | |
|-------|--------|------|------|------|------|------|------|-----|
| fmt 4 | 0x0087 | | | | sub | R1 | R0 | ↗73 |
| fmt 4 | 0x6543 | | | | mov | R2 | R1 | ↗74 |
| fmt 2 | nop | add | R15 | R0 | nop | | | ↗75 |

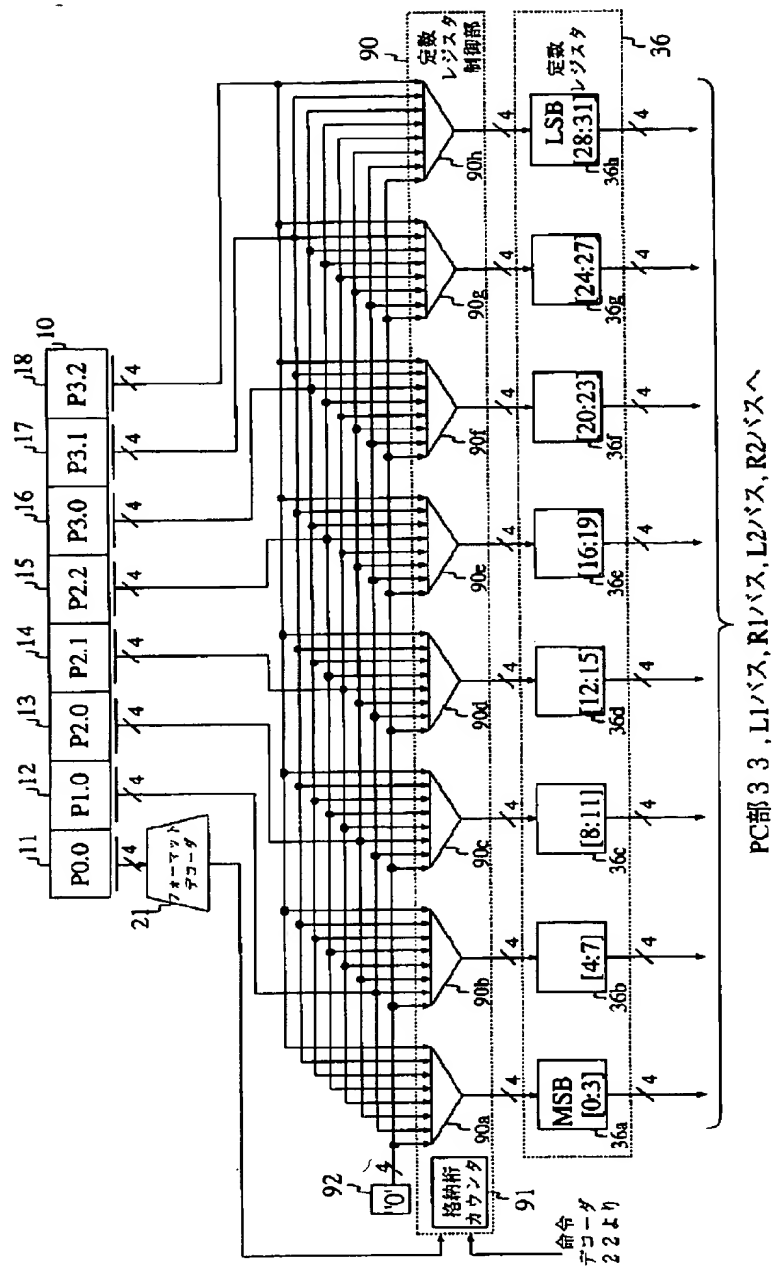
【図 1 2】

| P0.0 | P1.0 | P2.0 | P2.1 | P2.2 | P3.0 | P3.1 | P3.2 | |
|-------|--------|------|------|------|------|------|------|-----|
| fmt 4 | 0x1234 | | | | cmp | R0 | R2 | ↗76 |
| fmt 2 | eqi | add | R2 | R1 | cmp | R4 | R1 | ↗77 |
| fmt 2 | gti | add | R3 | R2 | mov | R2 | R0 | ↗78 |

【図 1 3】

| P0.0 | P1.0 | P2.0 | P2.1 | P2.2 | P3.0 | P3.1 | P3.2 | |
|-------|--------|------|------|------|-------|------|------|-----|
| fmt 4 | 0x3210 | | | | cmp | R0 | R2 | ↗79 |
| fmt 2 | eqi | cmp | R2 | R1 | setl4 | 0xc | — | ↗80 |
| fmt 2 | gti | add | R3 | R2 | mov | R2 | R0 | ↗81 |

【図14】



フロントページの続き

(72)発明者 田中 哲也
 大阪府門真市大字門真1006番地 松下電器
 産業株式会社内

(72)発明者 高山 秀一
 大阪府門真市大字門真1006番地 松下電器
 産業株式会社内

(22)

特開平 1 1 - 5 3 1 8 7

(72)発明者 小谷 謙介
大阪府門真市大字門真1006番地 松下電器
産業株式会社内

(72)発明者 宮地 信哉
大阪府門真市大字門真1006番地 松下電器
産業株式会社内